IMPLEMENTASI CHALLENGE RESPONSE AUTHENTICATION MECHANISM (CRAM) UNTUK KEAMANAN TRANSAKSI PERANGKAT IoT

Hendro FJ Lami[1], Stephanie I Pella[2]

[1,2] Universitas Nusa Cendana/Teknik Elektro Adi Sucipto
Email: h.lami@ staf.undana.ac.id,
Email:s.i.pella@ staf.undana.ac.id

Info Artikel

Histori Artikel:
Diterima Feb 28, 2021
Direvisi Mar 31, 2021
Disetujui Apr 24, 2021

ABSTRACT

This research aims to secure data transaction in Internet of Things (IoT)devices using the challenge-response authentication mechanism (CRAM). The research choose uses ESP 8266 and ESP 32 to develop the system for their ability to run micropython programming language. Using a random challenge to grant authentication protects the system from replay attack from intruders. In each authentication process, the client receives a 10 digit random number to be encrypted using a shared key and sent back to the server. The server then checks if the client posses the correct key by decrypting the encrypted challenge using the same shared key. Access is granted if the decryption result is equal to the original challenge.
Keywords: *CRAM, IoT, cryptography,esp8266, esp32*

ABSTRAK

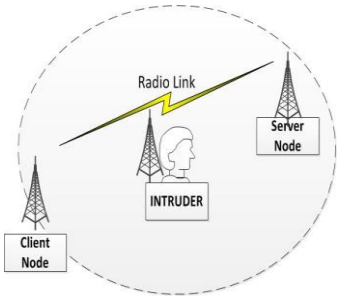
Penelitian ini bertujuan mengamankan transaksi data pada perangkat IoT menggunakan challenge response authentication mechanism (CRAM). Perangkat esp8266 dan esp32 menjadi pilihan karena perangkat vendor tersebut mampu diprogram menggunakan micropython. Melalui penggunaan random challenge untuk mendapatkan otentikasi, maka sistim menjadi lebih tangguh terhadap serangan intruder. Setiap meminta otentikasi client akan memperoleh maksimum 10 angka key yang harus dienkripsi dan dikirim ke sisi server. Pada sisi server akan melakukan dekripsi dan apabila hasil dekripsi mendapatkan data key yang sama maka otentikasi diberikan kepada client. Pada saat implementasi CRAM, client yang me miliki key berhasil melakukan enkripsi challenge dan berhasil kembali didekripsi oleh server sebagai syarat pemberian otentikasi.
Kata Kunci: *CRAM, IoT, cryptography,esp8266, esp32*

Penulis Korespondensi:
Hendro FJ Lami,
Program Studi Teknik Elektro Fakultas Sains dan Teknik,
Universitas Nusa Cendana,
Jl. Adisucipto Penfui - Kupang.
Email: h.lami@ staf.undana.ac.id

1. PENDAHULUAN

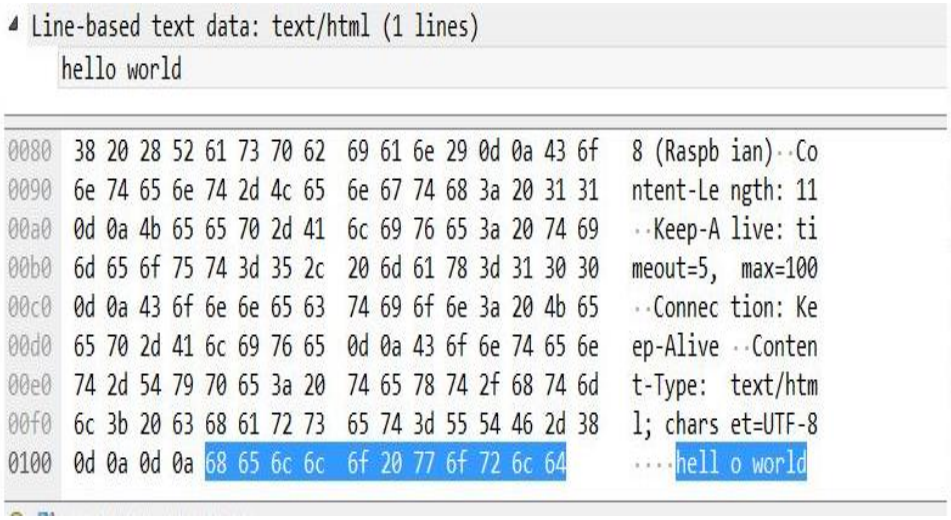
Percepatan teknologi internet of things (IoT) memberikan dampak perubahan pada beberapa aspek teknologi untuk membantu aktifitas manusia. Beberapa vendor pengembang perangkat IoT memacu inovasi dan produksi perangkat untuk mendukung aplikasi-aplikasi dalam bidang pertanian, keamanan, kesehatan, transportasi, dan lain sebagainya. Karakteristik sistim IoT ditentukan oleh kompleksitas jaringan pembentuk, serta kapabilitas perangkat lunak dan perangkat keras pendukung dalam memberikan solusi terhadap pencapaian tujuan[1], [2]. Beberapa vendor penyedia perangkat IoT berinovasi untuk menyediakan perangkat yang memiliki kapasitas dan kapabilitas sesuai keinginan pengembang. Raspberry pi mengeluarkan produk microcon-

troler bernama raspberry pico yang memiliki ukuran kecil [3], [4]. RobotDyn mengeluarkan produknya arduino mega 2560 R3-with-Wifi[5], dan espressif memproduski dua model board yaitu esp8266 dan esp32[6], [7]. Selain perangkat keras tersebut, perangkat lunak untuk proses pengembangan seperti arduino ide dan micropython mengalami percepatan guna pencapaian tujuan dalam disain aplikasi pada sisi sensor node maupun gateway[8]–[10]. Saat implementasi dalam suatu jaringan IoT, mikrokontroler-mikrokontroler tersebut berfungsi sebagai sensor node yang memiliki tugas memproses data hasil penginderaan untuk diteruskan ke gateway atau memproses tujuan penginderaan tertentu yang diminta oleh user. Keamanan transaksi data antara sensor node dan gateway ataupun transaksi data antar sensor node terhadap intruder ataupun hacker menjadi dasar pengembangan penelitian ini.



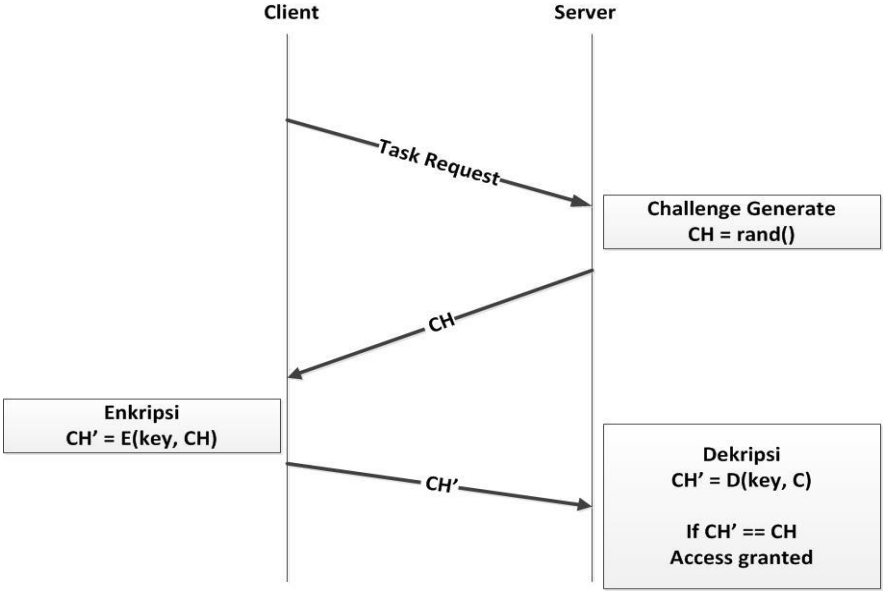
Gambar 1. Model Interfensi Intruder Pada Media Komunikasi Radio

Gambar 1. Menginformasikan bahwa user yang tak terdaftar dalam satu network mendapatkan informasi transaksi tanpa harus meminta hak akses terhadap data tersebut. Ketika intruder memiliki kemampuan untuk merekan aktivitas jaringan maka otomatis seluruh data dapat diketahui seperti terlihat pada gambar 2 termasuk username dan password. Kondisi ini potensial terjadi khususnya pada jaringan komunikasi radio yang umumnya menggunakan frekuensi yang sama dalam berkomunikasi[11]–[13]. Beberapa penelitian sebelumnya berhasil menggunakan perangkat espressif esp32 memodelkan proteksi data transaksi antar sensor node. Penelitian [14] memberikan kontribusi keamanan data perangkat espressif esp32 protokol MQTT menggunakan model username password. Pada[15] memberikan kontribusi keamanan SCADA pada sistem IoT dengan menggunakan protocol MQTT dan pemberian akses token. Selain kedua metode tersebut, [16] memberikan kontribusi bagaimana mengamankan data transaksi menggunakan model kriptografi.



Gambar 2. Potongan Data Perekaman Username dan Password Terenkripsi dan Data Tidak Terenkripsi Pada Protokol HTTP

Mempertimbangkan kemampuan intruder dalam mendapatkan data saat transaksi antar sensor node dan beberapa cara proteksi data transaksi maka pada penelitian ini akan memodelkan sebuah mekanisme yang dikenal dengan Challenge Response Authentication Mechanism (CRAM)[17]. Tujuannya untuk mengamankan informasi antar sensor node terhadap serangan interfensi data oleh intruder yang masuk kedalam sistem. Skenario CRAM antar client node dan server node terlihat pada gambar 3. Ketika client meminta server untuk melakukan suatu tugas tertentu maka server akan mengirimkan challenge ke sisi client untuk dienkripsi pada sisi tersebut. Data hasil enkripsi dikirimkan kembali ke sisi server untuk di lakukan dekripsi dan apabila sesuai maka client diberikan hak akses (granted) terhadap tugas tersebut.



Gambar 3 Challenge Request Authentication Mechanism (CRAM)

2. METODE PENELITIAN

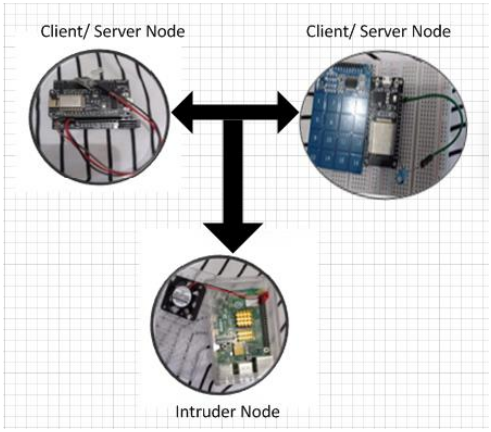
Penelitian ini menggunakan beberapa perangkat dan sensor yaitu antara lain :

Table 1 Perangkat Keras Pada Pemodelan CRAM

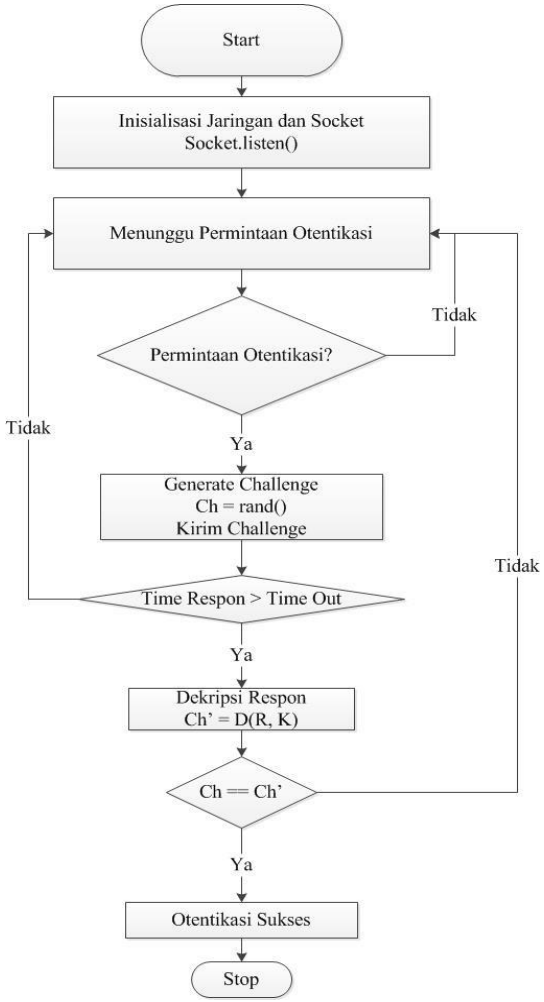
No	Nama Perangkat	Tujuan Penggunaan
1	Raspberry Pi 4/4GB	Monitor Sistem/ Client Node
2	Esp8266	Server Node/ Client Node
3	Esp32	Server Node/ Client Node

Selain perangkat lunak pada tabel 1, Penelitian ini membutuhkan bahasa program micropython untuk

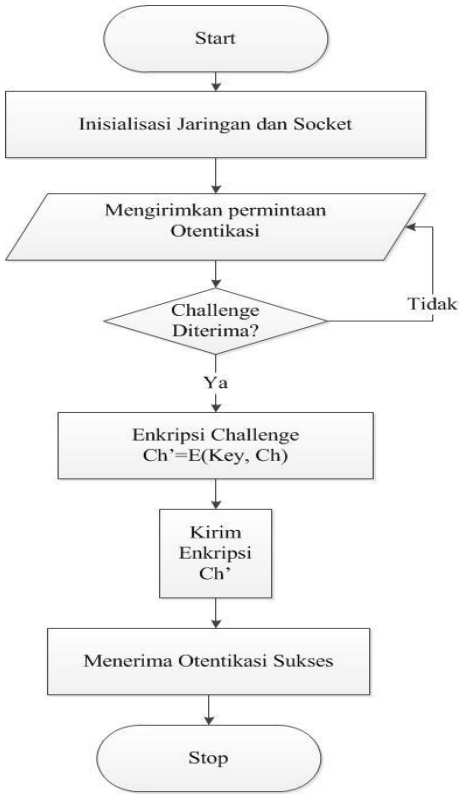
mempogram perangkat IoT guna pencapaian tujuan memodelkan CRAM. Skenario konfigurasi terlihat pada gambar 4 dan untuk mendukung keberhasilan tersebut maka terdapat beberapa tahapan dalam pemodelan seperti terlihat pada gambar 5 dan gambar 6. .



Gambar 4. Skenario Model CRAM



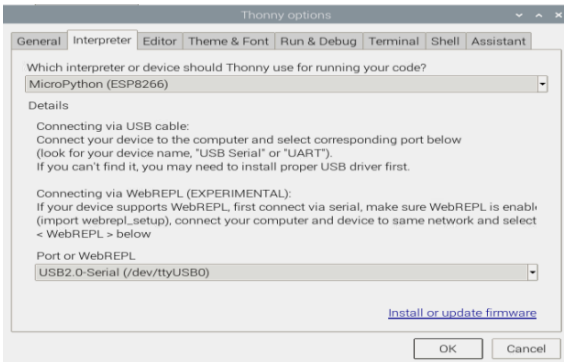
Gambar 5. Diagram Alir Server Node CRAM



Gambar 6. Diagram Alir Clien Node CRAM

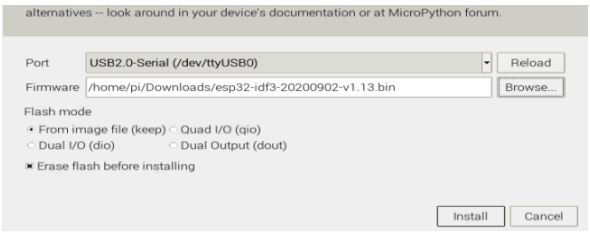
Terdapat dua firmware berbeda untuk masing-masing perangkat espressif tersebut dimana keduanya dapat diunduh pada[18]. Ada beberapa cara untuk upload firmware sehingga kedua board espressif tersebut dapat deprogram menggunakan micropython namun pada penelitian ini proses update firmware esp8266 dan esp32 melalui Thony editor yang terdapat pada sistim operasi raspbian buster. Tahapan-tahapan tersebut antara lain sebagai berikut:

1. Memilih perangkat (esp8266/esp32) dan port usb (/dev/ttyUSB0)

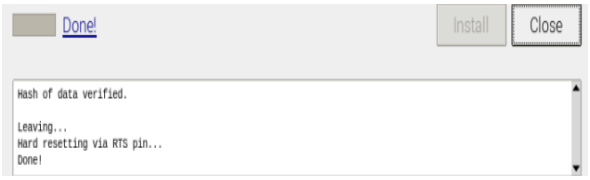


Gambar 7. Pemilihan Board dan Port untuk Update Firmware micropython

2. Install Firmware (esp8266 ataupun esp32)



Gambar 8. Upload Firmware ke Board Esp



Gambar 9. Instalasi Firmware Sukses

Ketika semua proses berjalan dengan baik tanpa ada nya tampilan error maka pada thonny editor akan menampilkan notifikasi pada shell bahwa editor tersebut siap utnuk memprogram board yang terinstall.

Gambar 10. Notifikasi Shell Perangkat Esp8266 dapat diprogram menggunakan micropython

Setelah proses instalasi firmware pada kedua board selesai maka tahapan selanjutnya adalah memprogram kedua board tersebut menggunakan bahasa micropython. Berikut ini adalah potongan kode program pada kedua perangkat tersebut, sebagai berikut :

1. Kode Program server-node inisialisasi jaringan dan port

```
ssid="
password="
port=5677
max_clients=1
host="192.168.1.3"

ap = network.WLAN(network.AP_IF)
ap.active(False)

print("Memulai Koneksi Ke Wifi '%s'...." % ssid)

wifi = network.WLAN(network.STA_IF)
wifi.active(True)
wifi.connect(ssid, password)
```

2. Kode Program socket.listen pada sisi server

```
if wifi.isconnected():
    print("Terkoneksi ke Wifi. IP Server: " + str(wifi.ifconfig()[0]))
    print("Next step is Starting server at port %d..." % port)

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sock.bind(('', port))
sock.listen(max_clients)
print("Status Server Aktif. Menunggu Client Terhubung!")
```

3. Kode Program Pembangkitan Challenge pada sisi server node

```
BLOCK_SIZE = 16
key = uos.urandom(32)
plaintext=str(urandom.getrandbits(32))
msg=plaintext
```

4. Kode Program Enkripsi Challenge pada sisi client node

```
cipher = aes(key, MODE_ECB)
jarak_spasi = BLOCK_SIZE - len(challenge) % BLOCK_SIZE
challenge = plaintext + " "*jarak_spasi
encrypted = cipher.encrypt(challenge)
print('AES-ECB encrypted:', encrypted )
cipher = aes(key,1)
```

5. Kode Program Dekripsi pada sisi server node

```
decrypted = cipher.decrypt(encrypted)
print('AES-ECB decrypted:', decrypted)
```

6. Kode Program Pengecekan dan Pengiriman Otentikasi Sukses

```
if encrypt=="decrypt":
    prin("Otentikasi Sukses, AKses Diberikan")
    print(host)
    print(port)
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    sock.connect((host, port))
    sock.send((msg.encode()))
```

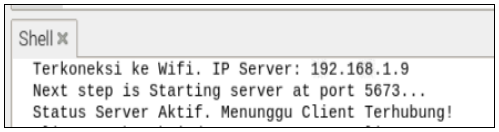
7. Kode Program visualisasi led pada server node saat kirim terima data

```
while True:
        data = conn.recv(1024).decode()
        print("Received from client: " + str(data))
        if data==encrypted:
            led = Pin(2,Pin.OUT)
            led.value(1)
            time.sleep(3)
            led.value(0)
            time.sleep(1)
```

3. HASIL DAN PEMBAHASAN

Setelah tahapan-tahapan konfigurasi, instalasi firmware kedua board, dan tahapan program selesai, maka tahapan selanjutnya adalah tahapan pengujian. Berikut ini tampilan dari beberapa hasil pengujian CRAM melalui shell monitor Thonny editor, sebagai berikut :

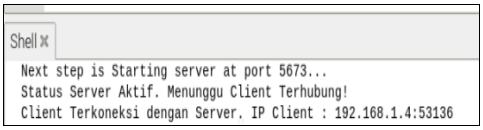
1. Inisialisasi Sukses Pada server



Gambar 11. Inisialisasi Koneksi Server Sukses

Gambar 10 memperlihatkan koneksi server ke sebuah jaringan wifi sukses dan telah mendapatkan Ip 192.168.1.9 dengan port komunikasinya adalah 5673.

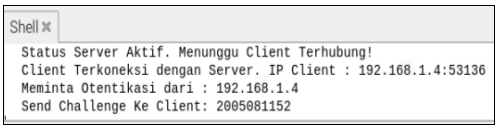
2. Client terkoneksi dengan server



Gambar 12. Client Node dan Server Node Terkoneksi

Terlihat pada gambar 12 sebuah client berhasil terkoneksi dengan server node pada Ip 192.168.1.4.

3. Challenge dikirimkan dari server ke client



Gambar 13. Pengiriman Challenge ke Client

Pada gambar 13, terlihat bahwa challenge terkirim dalam bentuk angka. Total karakter berjumlah 10 karakters. 10 karakter challenge tersebut berbeda setiap permintaan otentikasi oleh client node.

4. Client terima challenge dan mengirim enkripsi challenge ke server node.

Table 2. Data Challenge dan Challenge Terenkripsi

Challenge : 486103612 Enkripsi : \xa9\xfc\x15.\x13*\xd9\x95\xd6\xf1\xa6B\xb3\x8at\x7f4\ xa7+)\xcd\xbc\\,\x16\xf35\xfa8a\x8c
Challenge : 1950909101 Enkripsi : Z\x00\x8c\\\x007\xa9..\x9e\xde\x9d\xd1\xfc%\xa4t\x1a& \xff\xaa\xc4F]\xd3\x8f\x16\xb5\xb5q\xae(
Challenge : 3860304163 Enkripsi : %X\x9c\xc2\x97G"\xe54\xb3\xc4\xfa\x81V\xa7E\x8d\xf 433\x1a\x11\xa8\xcb\xae\x113\x196\x8b\x92Y
Challenge : 225957641 Enkripsi : \x8a/\xf3/{"\xe6$\x1bWZ\x92\xa8\xfa\xa9\x07\x96x{\xd8 \xb9\x10`\x1d\x1c\xa0\x01\xe8\x87s\xc0\xe9

5. Dekripsi enkripsi terkirim dari client pada sisi server

Table 3. Hasil Dekripsi Pada Server Node

Enkripsi : \xa9\xfc\x15.\x13*\xd9\x95\xd6\xf1\xa6B\xb3\x8at\x7f4\ xa7+)\xcd\xbc\\,\x16\xf35\xfa8a\x8c Dekripsi : 486103612
Enkripsi : Z\x00\x8c\\\x007\xa9..\x9e\xde\x9d\xd1\xfc%\xa4t\x1a& \xff\xaa\xc4F]\xd3\x8f\x16\xb5\xb5q\xae(Dekripsi : 1950909101
Enkripsi : %X\x9c\xc2\x97G"\xe54\xb3\xc4\xfa\x81V\xa7E\x8d\xf 433\x1a\x11\xa8\xcb\xae\x113\x196\x8b\x92Y Dekripsi : 3860304163
Enkripsi : \x8a/\xf3/{"\xe6$\x1bWZ\x92\xa8\xfa\xa9\x07\x96x{\xd8 \xb9\x10`\x1d\x1c\xa0\x01\xe8\x87s\xc0\xe9 Dekripsi : 225957641

Berdasarkan tabel 2 dan 3, pemberian otentikasi kepada client terjadi saat challenge yang terkirim menuju client berhasil didekripsi oleh server dengan hasil dekripsinya adalah data challenge yang diberikan oleh server node tersebut. Pembangkitan plain text dan key seperti yang terlihat pada kode program 3 dan kode program 4 menjadikan sistim lebih tangguh terhadap serangan karena data challenge dan key menjadi dinamis.

4. KESIMPULAN

Sistim CRAM merupakan cara mengamankan data perangkat IoT terhadap serangan intruder. Pembangkitan challenge secara random menyebabkan sistim lebih dinamis karena data enkripsi berubah untuk tiap permintaan otentikasi. Penggunaan maksimum 32 bit pada pembangkitan data random menghasilkan maksimum 10 angka challenge random tiap permintaan otentikasi. Kondisi ini sangat berguna bagi sistim IoT saat menerima serangan pengambilan data otentikasi.

DAFTAR PUSTAKA

[1] V. D. Soni, "Security issues in using iot enabled devices and their Impact," *Int. Eng. J. Res. Dev.*, vol. 4, no. 2, p. 7, 2019.

[2] K. Tabassum, A. Ibrahim, and S. A. El Rahman, "Security issues and challenges in IoT," in *2019 International Conference on Computer and Information Sciences (ICCIS)*, 2019, pp. 1–5.

[3] T. R. P. Foundation, "Buy a Raspberry Pi Pico," *Raspberry Pi*. https://www.raspberrypi.org/products/raspberry-pi-pico/ (accessed Feb. 21, 2021).

[4] "raspi pico pdf - Penelusuran Google." https://www.google.com/search?q=raspi+pico+pdf&rlz=1C1PRFC_enID911ID911&oq=ras-

pi+pico+pdf&aqs=chrome..69i57j0i19i22i30l6j69i61.8533j0j7&sourceid=chrome&ie=UTF-8 (accessed Feb. 21, 2021).

[5] "MEGA+WiFi R3 ATmega2560+ESP8266, flash 32MB, USB-TTL CH340G, Micro-USB." https://robotdyn.com/mega-wifi-r3-atmega2560-esp8266-flash-32mb-usb-ttl-ch340g-micro-usb.html (accessed Feb. 21, 2021).

[6] M. Babiuch, P. Foltýnek, and P. Smutný, "Using the ESP32 microcontroller for data processing," in *2019 20th International Carpathian Control Conference (ICCC)*, 2019, pp. 1–6.

[7] H. F. Lami, K. R. Rantelobo, J. F. Mandala, and A. S. Sampeallo, "INTEGRASI PROTOKOL MQTT DAN HTTP UNTUK OTOMASI BERBASIS IOT PADA PERTANIAN LAHAN KERING," *J. Media Elektro*, pp. 53–59, 2020.

[8] "Software." https://www.arduino.cc/en/software (accessed Feb. 21, 2021).

[9] C. Bell, *MicroPython for the Internet of Things*. Springer, 2017.

[10] "MicroPython - Python for microcontrollers." http://micropython.org/ (accessed Feb. 21, 2021).

[11] J. Li, K. Fawaz, and Y. Kim, "Velody: Nonlinear vibration challenge-response for resilient user authentication," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 1201–1213.

[12] A. Vyas and S. Pal, "Preventing security and privacy attacks in WBANs," in *Handbook of computer networks and cyber security*, Springer, 2020, pp. 201–225.

[13] A. Ometov, V. Petrov, S. Bezzateev, S. Andreev, Y. Koucheryavy, and M. Gerla, "Challenges of multi-factor authentication for securing advanced IoT applications," *IEEE Netw.*, vol. 33, no. 2, pp. 82–88, 2019.

[14] C. R. Aldawira, H. W. Putra, N. Hanafiah, S. Surjarwo, and A. Wibisurya, "Door security system for home monitoring based on ESp32," *Procedia Comput. Sci.*, vol. 157, pp. 673–682, 2019.

[15] L. O. Aghenta and T. Iqbal, "Design and implementation of a low-cost, open source IoT-based SCADA system using ESP32 with OLED, ThingsBoard and MQTT protocol," *AIMS Electron. Electr. Eng.*, vol. 4, no. 1, pp. 57–86, 2019.

[16] M. Suárez-Albela, P. Fraga-Lamas, L. Castedo, and T. M. Fernández-Caramés, "Clock frequency impact on the performance of high-security cryptographic cipher suites for energy-efficient resource-constrained IoT devices," *Sensors*, vol. 19, no. 1, p. 15, 2019.

[17] P. Kushwaha, H. Sonkar, F. Altaf, and S. Maity, "A Brief Survey of Challenge–Response Authentication Mechanisms," in *ICT Analysis and Applications*, Springer, 2021, pp. 573–581.

[18] "MicroPython - Python for microcontrollers." http://micropython.org/ (accessed Feb. 24, 2021).